Ecology, 75(5), 1994, pp. 1289–1298

CONSTRUCTION OF MULTIDIMENSIONAL CLUSTERED PATTERNS[1]

MARC MANGEL AND FREDERICK R. ADLER[2]
Section of Evolution and Ecology and Center for Population Biology, University of California, Davis, California 95616 USA

Abstract. Ecological processes often depend upon the patterning, as well as the absolute density, of resources. In this paper, we develop methods for describing pattern from the perspective of the organism encountering and exploiting the resources, and for reconstructing pattern from the description. The essence of our description is the "structure function," which is the probability that a point r units away from the current point contains resources, conditional on the resource state of the current point. We first show how the structure function is determined from pattern and then describe an algorithm (the method of the "Force to be Full") for constructing pattern in any number of dimensions from a given structure function. We illustrate our ideas with empirical data from krill surveys and with simulated but complex three-dimensional patterns.

Key words: algorithms; clustered resources; foraging; krill; spatial pattern.

THE ROLE OF PATTERN IN DESCRIPTIVE AND PREDICTIVE ECOLOGY

Many ecological processes depend on the spatial pattern of resources or organisms and not just on their density. Regularly distributed food items might be more difficult than clumped food items for a forager to find. Regularly distributed sessile competitors might develop a less exaggerated size hierarchy than random or clumped competitors. Regularly distributed flowers might have lower pollination success than flowers in clusters.

Distinguishing regular patterns from clustered patterns is a nontrivial task. Most commonly, a variety of spatial statistics are used to distinguish patterns from a random pattern, with the hope of deducing something about the process that produced the pattern (Cressie 1991). In other cases, a set of descriptors or statistics about pattern may be used to deduce the consequences of pattern. Here we propose a description of pattern based on the experience of an organism encountering that pattern. Because of a focus on the organism in its ecological setting, this description can be used in a predictive way (Mangel 1994).

For a description of this sort to be useful, it must be possible to translate it back into pattern. To this end, we describe an algorithm for generating patterns matching a given description. Such a construction allows for design of field and computer experiments that control spatial pattern in a meaningful way.

The description of space and spatial pattern has fundamental consequences for reasoning about and modeling spatial processes. For example, an essentially continuous view of space underlies the large class of diffusion models. With this sort of model, clumpy environments, usually produced by discrete individuals or resources, are difficult to describe and tend to be ignored. Similarly, such models generally have dynamics based upon strictly local interactions, an often inappropriate restriction that has fundamental dynamic consequences (but see Levin and Segel 1985). Cellular automata, in which the space is discretized and the state of a particular cell depends upon the states of neighboring cells, can include nonlocal interactions, but generally assume a very regular geometry of sites (such as a regular square or hexagonal lattice) with unknown implications for dynamics. Such methods overly restrict the assumptions about the nature of dynamics.

On the other hand, many spatial statistics give no insight into process. Nearest neighbor statistics, for example, ignore all global aspects of the pattern. Semivariograms and related methods, although similar to the treatment of space introduced in this paper, are highly phenomenological in their description of pattern, combining the data from different locations without a mechanistic biological underpinning. The approach that we use in this paper allows one to provide information about resources at all distances.

Our goal is to avoid the dangers of overly restrictive assumptions about the structure of space and of overly general descriptions of space. By beginning from the perspective of the organism moving in space, we produce a general description of spatial pattern. This description is neither so detailed as to be consistent with only a single pattern, nor so general as to be an insufficient basis for the construction of patterns.

Our new tool for describing and constructing pattern is the "structure function," which is the chance that a

[1] Manuscript received 9 April 1993; revised 10 October 1993; accepted 20 October 1993; final version received 18 November 1993.

[2] Present address: Departments of Biology and Mathematics, University of Utah, Salt Lake City, Utah 84112 USA.

spatial point some distance away from a given point contains resources, conditioned on the resource state at the current point. Using the simple first-order Markov process in one dimension, we illustrate the processes of translating from pattern to structure function and from structure function to pattern. With this basis, we proceed to the more difficult and interesting multidimensional cases, showing how to compute the structure function from different sorts of empirical or simulated data. We conclude by presenting the method of the "Force to be Full," an algorithm to produce patterns matching a given structure function in multiple dimensions, and illustrate the technique with both simulated and empirical data.

The Structure Function: Describing Pattern in a Meaningful Way

An organism living in a world with spatially structured resources can attend to two kinds of information, global and local. Global information consists of the overall density of resources and ignores spatial pattern. Local information includes some description of that pattern. A complete description is a map of the location of every item. There are many drawbacks to storing information in this way, and a less complete description might be preferable. For example, a forager might want to know, given that the current locale has or does not have resource, what the likelihood is that there is resource in the vicinity of a point r units away. This local information constitutes the structure function, which is the central feature of our analysis:

$$p(r|1) = \text{Prob}\{\text{resource in the vicinity of a point } r \text{ units away, given that there is resource at the current point}\}$$
$$p(r|0) = \text{Prob}\{\text{resource in the vicinity of a point } r \text{ units away, given that there is no resource at the current point}\}. \quad (1)$$

To start, we note that the two aspects of the structure function are connected. If p_a is the average density of resource in the environment, then

$$p_a p(r|1) + (1 - p_a)p(r|0) = p_a. \quad (2)$$

The left-hand side gives the probability of finding resources at a point a distance r from a randomly chosen point. That is, if the organism starts at a point with resources (with probability p_a) and moves a distance r, the probability is $p(r|1)$ that the new point has resources. If it starts at a point without resource [with probability $(1 - p_a)$], the probability that the point r units away has resource is $p(r|0)$. Together, these must give the average density p_a of resources in the environment.

Solving Eq. 2 for $p(r|0)$ gives

$$p(r|0) = \frac{p_a[1 - p(r|1)]}{1 - p_a}. \quad (3)$$

Thus, it is sufficient to measure p_a and construct $p(r|1)$, since $p(r|0)$ can be obtained from it.

The structure function is related to the common description of pattern by the semivariogram (Mackas et al. 1985). To see this, let $Z(r) = 1$ if there is resource in the vicinity of the point r, let $Z(r) = 0$ otherwise and let d measure distance from the randomly picked point at r. The semivariogram is

$$\mathbf{S}(d) = 0.5E\{Z(r) - Z(r + d)\}^2. \quad (4)$$

To compute Eq. 4, we condition on $Z(r) = 1$ (with probability p_a) or $Z(r) = 0$ (with probability $1 - p_a$). Then

$$\mathbf{S}(d) = 0.5\{p_a[1 - p(d|1)] + (1 - p_a)p(d|0)\}. \quad (5)$$

The first term on the right-hand side arises as follows: if $Z(r) = 1$, then $Z(r + d) = 1$ with probability $p(d|1)$ and $\{Z(r) - Z(r + d)\}^2 = 0$; similarly $Z(r + d) = 0$ with probability $1 - p(d|1)$ and $\{Z(r) - Z(r + d)\}^2 = 1$. The second term is derived by a similar argument. The advantage of the structure function over the semivariogram can be seen by comparing Eq. 5 and Eq. 1. Both provide information about how rapidly the environmental average is approached from a local point. The structure function retains local information (about the resource state of the current point), whereas the semivariogram loses this information through the averaging.

Generating the Structure Function from Pattern and Vice Versa

In this section, we show how to translate from the structure function to the pattern and from the pattern to the structure function. We focus on clustered patterns of resource (rather than regular patterns), but the general method of the structure function can be used to study regular patterns as well. We begin with a simple and instructive one-dimensional case to clarify ideas, although our main focus is two or three dimensions. When one-dimensional pattern is generated by a first-order Markov process, it is possible both to compute the structure function analytically and to produce matching patterns. In all other cases, this simplicity is lost. We then discuss how to determine structure functions from empirical data either directly (when data are given completely) or computationally using an approximate method (when only nearest neighbor distances are available). Finally, we provide an algorithm (the method of the "Force to be Full") for creating spatial pattern in two or three dimensions, given a structure function.

One-dimensional, first-order Markov patterns and the associated structure functions

When pattern is generated by a first-order Markov process, the probability that a particular location has resource depends only on the status of a single neighbor. That is, one can think of the environment as a

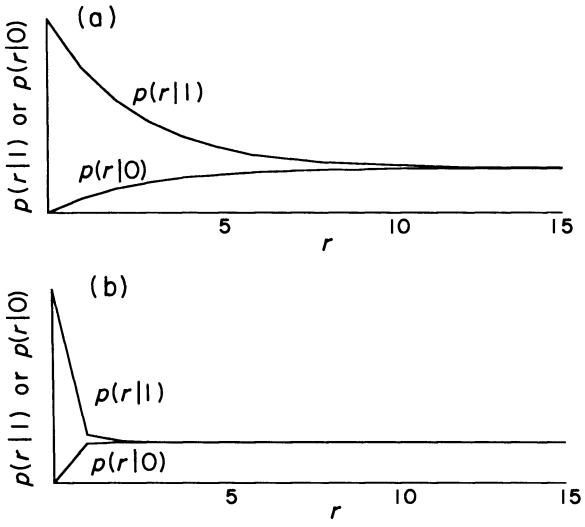

FIG. 1. Examples of structure functions generated by a one-dimensional Markov process. (a) Parameters $p_1 = 0.75$, $p_0 = 0.075$ so that $p_a = 0.231$; (b) parameters $p_1 = 0.25$, $p_0 = 0.2$ so that $p_a = 0.211$. Variables defined at Eqs. 1 and 2.

series, with the probability of resource at a given point depending only on the state of its left neighbor. This makes construction of such patterns simple. If the spatial structure is generated in this way, it is also simple to calculate the structure function.

The first-order Markov process is fully specified by transition probabilities p_1 and p_0 representing, respectively, the probabilities of finding resource at a site immediately adjacent to a full and to an empty site. Setting $Z(r)$ to be 1 in the presence of resource and 0 in its absence as before, we have

$$p_1 = \Pr\{Z(r+1) = 1 \mid Z(r) = 1\}$$
$$p_0 = \Pr\{Z(r+1) = 1 \mid Z(r) = 0\}. \quad (6)$$

Note that p_1 and p_0 are related by a condition similar to Eq. 2

$$p_a p_1 + (1 - p_a)p_0 = p_a, \quad (7)$$

and thus that two of p_a, p_1, and p_0 determine the third.

Generating such a pattern is straightforward. Beginning from a point with resources at location 0, establish the status of subsequent points consecutively based on the status of adjacent points and the probabilities p_1 and p_0. For example, to set the state of point $i + 1$, choose a uniform random number between 0 and 1. If there is resource at site i, place resource in site $i + 1$ if the number is less than p_1, using p_0 in a parallel way if there is no resource at site i.

Computing the structure function in this case is possible analytically. Suppose we are at a point with resource, so that $Z(0) = 1$, and wish to compute the probability $p(r \mid 1)$ that a point r units away contains resource. This can happen in two ways. If $Z(r-1) = 1$ [with probability $p(r-1 \mid 1)$] then point r has resource with probability p_1. If $Z(r-1) = 0$ [with probability $1 - p(r-1 \mid 1)$] then point r has resource with probability p_0. Putting these together, we have

$$p(r \mid 1) = p_1 p(r-1 \mid 1) + p_0\{1 - p(r-1 \mid 1)\}$$
$$= p_0 + [p_1 - p_0]p(r-1 \mid 1). \quad (8)$$

Subtracting p_a from both sides, and using Eq. 7, we find that

$$p(r \mid 1) - p_a = [p_1 - p_0][p(r-1 \mid 1) - p_a].$$

With initial condition $p(1 \mid 1) = p_1$, the solution of this recursive equation is

$$p(r \mid 1) = p_a + (1 - p_a)[p_1 - p_0]^r. \quad (9)$$

The structure function thus depends geometrically on the transition probabilities p_1 and p_0 (Fig. 1).

Structure functions associated with the Markov process with different parameters give a hint of how pattern may affect behavior. An organism moving in the world described by Fig. 1a, where resources tend to be tightly clustered, will most likely behave quite differently from an organism moving in the world characterized by Fig. 1b, where resources are more or less randomly distributed. We might predict long distance moves from a point devoid of resources to search for resource clusters in the first case, but not in the second one. The structure function thus illustrates the logic underlying area-restricted search (Bell 1991).

Constructing structure functions from empirical and computer-generated data

It is common that we do not know the stochastic process that generates a particular pattern. We here show how to estimate the structure function from empirical data. First, we convert nearest neighbor distances to resources in one dimension into a structure function and recommend a related procedure for collecting data to make possible estimation of the structure function in multiple dimensions. We conclude by showing how to convert a complete map of the resource distribution into a structure function in any dimension.

When space is measured discretely, the nearest neighbor distribution takes the form

$$g(r) = \text{Prob}\{\text{nearest neighbor to a given resource is } r \text{ units away}\}. \quad (10)$$

Since this is a discrete distribution, $\sum_{r=1}^{\infty} g(r) = 1$.

In one dimension, we can use the nearest neighbor distance to recreate the structure function, if we make the additional assumption that the process generating the pattern is similar to the first-order Markov case. Imagine creating a distribution of resources sequentially, by picking an initial resource point, moving a distance r with probability $g(r)$ to the next resource

point, and then repeating the process from the new point. The additional assumption encoded in this algorithm is the independent choice of r in each step. (The alternative is that the nearest neighbor distances of consecutive resources are correlated.)

This construction allows for computation of the structure function in a way analogous to the first-order Markov case. As before, suppose we are at a point with resource and wish to compute the probability $p(r|1)$ that a point r units away contains resource. There are once again two cases. If the nearest neighbor distance d from the first point is greater than r, then point r definitely has no resource. If the nearest neighbor distance d is less than or equal to r, then point r is occupied with probability $p(r - d|1)$. Note that the case $d = r$ is covered, because $p(0|1) = 1$. Adding these terms together gives

$$p(r|1) = \sum_{d=1}^{r} g(d)p(r - d|1), \qquad (11)$$

which can be solved iteratively for $p(r|1)$ given $\{g(d), d = 1, 2, \ldots\}$. In Appendix 1 we show how this case generalizes the first-order Markov case.

A form of nearest neighbor distance can be used to estimate structure functions in the field. To be specific, suppose that we want to characterize spatial distribution of fruit, say rose hips, in bushes. To estimate the structure function without mapping all the fruit, one can go to a number of bushes and randomly pick a "central fruit" in each. From this central fruit, extend a number of rays picked with random orientation and measure the presence or absence of fruit at various distances along each of these rays. For example, if fruit were found in the distance range 40–50 cm in 6 out of 20 cases, $p(50|1)$ would be estimated as 6/20, or 0.3.

When a complete map is given (e.g., Casas 1990), it is easy to measure the structure function. To do this, let x_1 and x_2 denote any two spatial points, let $d(x_1, x_2)$ denote the distance between them, and let

$$I[r, d(x_1, x_2)] = \begin{cases} 1 & \text{if } r = d(x_1, x_2) \\ 0 & \text{otherwise.} \end{cases} \qquad (12)$$

Thus, $I[r, d(x_1, x_2)]$ "indicates" whether ($I = 1$) or not ($I = 0$) the distance between the two points is exactly r. The structure function can be computed with the following algorithm. To begin, specify a region X, which is far away from the boundaries of the map.

Algorithm 1: constructing the structure function from a pattern.—

1) Cycle over all x_1 in X.

2) If $Z(x_1) = 1$ (so that there is resource at x_1), then cycle over all x_2.

3) Set

$$p_{x1}(r|1) = \sum_{x_1} Z(x_1)Z(x_2)I[r, d(x_1, x_2)] / \sum_{x_1} I[r, d(x_1, X_2)].$$

4) Average the p_{x1}.

This algorithm computes the structure function for a given resource distribution in any number of dimensions.

Generating two- and three-dimensional patterns from the structure function: the method of the "Force to be Full"

We have shown how to compute structure functions from pattern in any number of dimensions, but have only shown how to construct patterns from Markovian structure functions and nearest neighbor distances, and that only in one dimension. These methods fail in higher dimensions because locations are not ordered, that is, in two or three spatial dimensions there is no "next" point to be generated by a Markov process or by a nearest neighbor distribution. (This is even true in one spatial dimension, in which case we have to make an arbitrary decision about whether the nearest neighbor to the left or the right is the one that determines the resource state of the current cell.) A more global approach to generating pattern is thus required.

We now describe an algorithm for the generation of a two- or three-dimensional spatial pattern to match a given structure function; it is designed to iteratively approach an appropriate pattern. At each step, resources are generated at a given point according to whether or not the point experiences a large "force to be full" from the rest of the pattern. Each point in the pattern, whether empty or full, has an effect on whether or not the given point should be full, expressed by the structure function and the distance to the point. An appropriate combination of these effects is the force to be full. Points with a large force to be full are filled with high probability, those with a small force to be full are filled with low probability. The algorithm then continues with the new pattern until the measured structure function (Algorithm 1) closely matches the target structure function.

We let x denote a particular spatial location in the range $0 \le x \le x_{max}$, with x representing either one, two, or three dimensions. We let n represent the iterate, and $Z_n(x)$ describe the pattern at the n^{th} iterate, being equal to 1 when there are resources at x, and 0 otherwise. Set the maximum number of iterates to be n_{max}. This algorithm is sufficiently complex that we follow several steps with interpretation.

Algorithm: the Force to be Full.—

1) Set $n = 0$, and create the "zeroth" spatial pattern by filling points independently to match the average density of resource p_a. That is, cycle over all x and at each point choose a number u randomly distributed on 0 to 1 and set $Z_0(x) = 1$ if $u < p_a$ and $Z_0(x) = 0$ otherwise.

Interpretation: Lacking a better starting point, we randomly distribute resources in space, so that the average probability of a point containing resources is p_a, determined from the structure function.

2) Let "1" denote all the points x where $Z_n(x) = 1$ and let "0" denote all the points where $Z_n(x) = 0$.

3) Cycle over all x. At each spatial point x, cycle over all spatial points x' and let $d(x; x')$ denote the distance between x and x'. Set $K[d(x; x')] = \exp[-\beta d(x; x')]$, where β is a user-provided parameter (see *Examples* below) and set

$$G_1(x) = \sum_{x' \in \text{"1"}} p[d(x; x')|1]K[d(x; x')] + \sum_{x' \in \text{"0"}} p[d(x; x')|0]K[d(x; x')] \quad (13)$$

and

$$G_0(x) = \sum_{x' \in \text{"1"}} \{1 - p[d(x; x')|1]\}K[d(x; x')] + \sum_{x' \in \text{"0"}} \{1 - p[d(x; x')|0]\}K[d(x; x')]. \quad (14)$$

Interpretation: The function $K(d)$ measures the "influence" of the presence or absence of resources at one point on the presence or absence of resources at another point. We choose an exponentially decaying function for simplicity. Some numerical experimentation is required to determine an appropriate value of β. For example, if β is "large," say >1, then essentially only nearest neighbors will influence a point. On the other hand, if $\beta = 0$, then all points equally influence each other. In the computations reported in the next section, we use $\beta = 0.25$.

Think of $p[d(x; x')|1]$ as the "desire" of a full point x' that point x shares its good fortune in having resources and of $p[d(x; x')|0]$ as the desire of an empty point x' that point x enjoys resources. Then $G_1(x)$ measures the summed desire of all spatial points that point x contains resources. Similarly, $G_0(x)$ measures the summed desire of all spatial points that point x lacks resources. Alternatively, we can envision a "flow" of resources from one filled point towards an empty point. Then $G_1(x)$ measures the summed flow from all filled points to point x.

4) The "force to be filled" at x is

$$FF(x) = \frac{G_1(x)}{G_1(x) + G_0(x)}. \quad (15)$$

Interpretation: This function compares G_1 and G_0. If G_1 is much larger than G_0 the point x has a large force to be full.

We shall now rescale the force to be filled twice. In the first rescaling, we expand the values of $FF(x)$ to cover 0 to 1. In the second scaling we ensure that the average value is p_a.

5) To rescale $FF(x)$, let $F_{\min}$ and $F_{\max}$ be the smallest and largest values of $FF(x)$ over all values of x, and set

$$FF'(x) = \frac{FF(x) - F_{\min}}{F_{\max} - F_{\min}}. \quad (16)$$

Next determine the parameter q so that

$$\frac{1}{x_{\max}} \sum_x [FF'(x)]^q = p_a \quad (17)$$

and set

$$FF''(x) = [FF'(x)]^q. \quad (18)$$

Interpretation: Eq. 16 linearly rescales the force to be full to cover the range from 0 to 1. Eq. 17 preserves this range and is solved for q to simultaneously ensure that the average doubly rescaled force to be full, FF'' as defined by Eq. 18, is p_a. (A nonlinear scaling is required for the algorithm to work; we don't know why this is so. In general, q is substantially different from 1.)

Eq. 17 is a nonlinear equation for q. It can be solved effectively by Newton's method (Press et al. 1986). We have not yet encountered a case in which the Newton method did not converge in 10 or fewer iterations.

6) Cycle over all x setting $Z_{n+1}(x) = 1$ if a uniformly distributed random number u is less than $FF''(x)$ and setting $Z_{n+1}(x) = 0$ otherwise. Because the average value of FF'' is p_a, the expected density of resources is also p_a.

7) Construct the empirical structure function using Algorithm 1. Set the empirical structure function to be $SF_{n+1}(d)$.

8) Construct the goodness of fit between the given structure function $p(r|1)$ and the empirical structure function by

$$D_{n+1} = \sum_{r=0}^{x_{\max}} |p(r|1) - SF_{n+1}|\omega(r), \quad (19)$$

where $\omega(r)$ is the "weighting" assigned to distance r in measuring the goodness of fit. For example, if $\omega(r) = 1$, then all distances are weighted equally whereas if $\omega(r)$ decreases as r increases, then small distances "count more" in the measure of the goodness of fit. If the value of D_{n+1} so constructed is smaller than that of the previous best pattern (i.e., with the previous smallest D_n), then store the current pattern as the best pattern and the current goodness of fit as the best goodness of fit.

9) Replace the current value of n by $n + 1$. If the new value is less than $n_{\max}$ then return to step 2. Otherwise, stop and adopt the pattern with the smallest value of D_n. This algorithm does not "converge," in the sense that as n increases D_n decreases. However, if enough iterations are considered, an excellent fit between the empirical and underlying structure functions can be obtained.

Examples

We shall now illustrate how some of these ideas can be used to generate structure function from pattern and

TABLE 1. Nearest neighbor frequency distribution for krill swarms, reported by Butterworth et al. (1991), based on the data of Miller and Hampton (1989).

Distance interval (km)	Frequency of swarms
0–0.2	328
0.2–0.4	277
0.4–0.6	162
0.6–0.8	122
0.8–1.0	86
1.0–1.2	69
1.2–1.4	49
1.4–1.6	42
1.6–1.8	26
1.8–2.0	35
2.0–3.0*	101
3.0–4.0	54
4.0–5.0	28
>5.0	187

* We assume that swarms in the 2.0–3.0, 3.0–4.0, and 4.0–5.0 km intervals are uniformly distributed in those intervals and that swarms in the >5.0 category were uniformly distributed between 5.0 and an upper limit of 15.0 km.

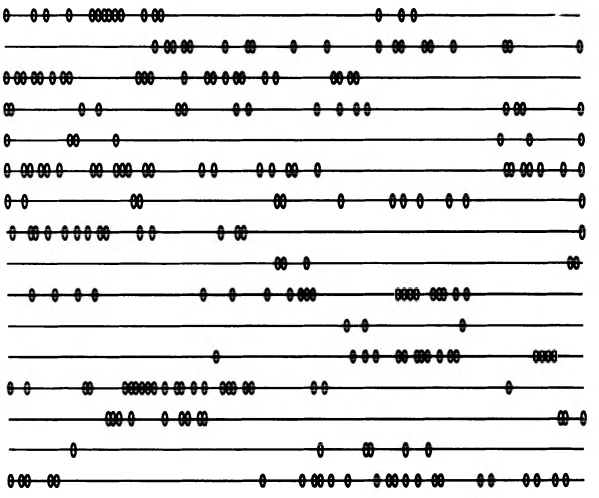

FIG. 2. A "recreated" linear transect through krill aggregations, using the nearest neighbor distances from Table 1 to construct the spatial pattern using Algorithm 1.

pattern from structure function. The computations described here were run on a Macintosh IIfx, Quadra 700, or Quadra 800 using TRUEBASIC. Because each iteration of the Force to Be Full algorithm requires repeated cycling over all spatial points twice, one is likely to run out of time and patience waiting for the result before one runs out of computer memory. In Appendix 4, we discuss the timing of the algorithm in more detail.

Butterworth et al. (1991) published the nearest neighbor distances (Table 1) for krill aggregations studied by Miller and Hampton (1989). In this study, ≈1500 krill (*Euphausia superba*) aggregations were detected and sized acoustically by cruising linear transects in an area of the southwest Indian ocean during the First International Biomass Experiment (FIBEX).

In order to achieve a consistent spatial increment in the construction of the structure function, we made the additional assumption (see Table 1) that the swarms in the 2.0–3.0, 3.0–4.0, and 4.0–5.0 km intervals were uniformly distributed in those intervals and that the swarms in the >5.0 km category were uniformly spread between 5.0 km and an arbitrary upper limit of 15.0 km (the actual choice had little effect on the structure function in this case). Thus, the distribution of nearest neighbor distances has maximum 15 km. Consequently Eq. 10 is normalized with a maximum value of $r = 15$. With these modifications, we can recreate a sample path of the kind measured by Miller and Hampton (Fig. 2) and the structure function (Fig. 3). This essentially one-dimensional pattern appears to be generated by something close to a first-order Markov process.

Higher order Markov processes, even in one dimension, incorporate spatial correlations, in that the state of a point depends on several of its neighbors to the left and not just the first. A second-order Markov process requires that we define

$$p_{ij} = \text{Prob}\{Z(r) = 1 \mid Z(r-1) = i,\ Z(r-2) = j\} \quad (20)$$

for $i = 0$ or 1 and $j = 0$ or 1. For example p_{01} is the probability that a site is filled when its immediate neighbor to the left is empty, but its next neighbor is full. These values generalize p_0 and p_1 in Eq. 6. Similarly, description of a third-order Markov process requires values for

$$p_{ijk} = \text{Prob}\{Z(r) = 1 \mid Z(r-1) = i,\ Z(r-2) = j,\ Z(r-3) = k\} \quad (21)$$

for $i = 0$ or 1, $j = 0$ or 1, and $k = 0$ or 1. Generation of patterns follows an algorithm similar to that for first-order Markov processes and is described in Appendix 2.

We challenged the "Force to be Full" algorithm to produce three-dimensional patterns to match the structure function from a variety of third-order Markov processes. The parameters $p_{000} = 0.368$, $p_{010} = 0.444$, $p_{001} = 0.273$, $p_{011} = 0.473$, $p_{100} = 0.21$, $p_{101} = 0.045$,

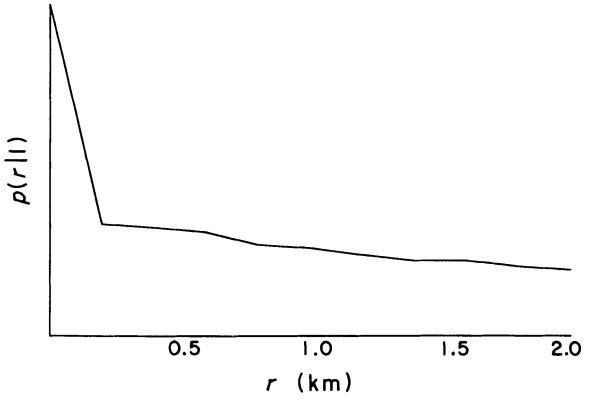

FIG. 3. The structure function for krill aggregations based on the recreated transect in Fig. 2.

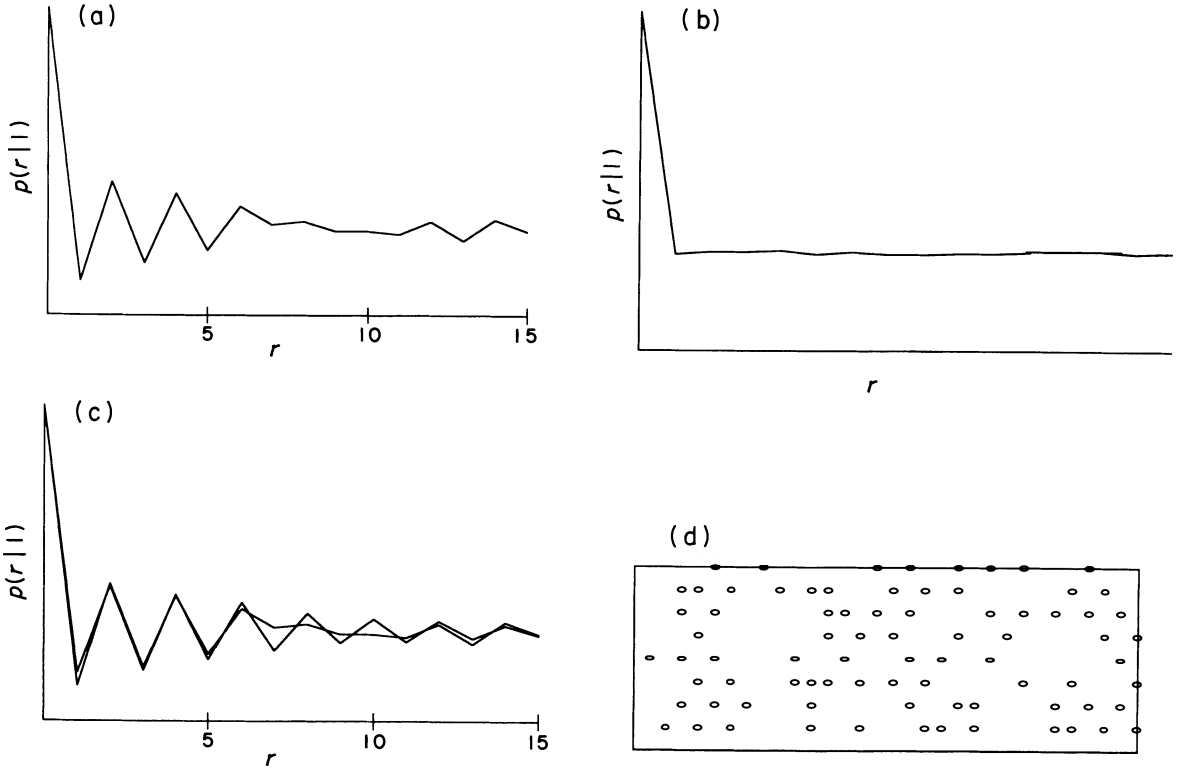

FIG. 4. (a) The structure function generated by the third-order Markov process described in the text. (b) The empirical structure function generated during the first iteration (i.e., random allocation of resources over space) of the "Force to be Full" algorithm. (c) The best empirical structure function, taken over 20 iterations of the "Force to be Full" algorithm, compared with the structure function generating it. (d) A cross section, at $z = 4$, of this empirical pattern.

$p_{110} = 0.014$, $p_{111} = 0.217$ produce a structure function (Fig. 4a) with many "peaks and valleys." We applied the method of the force to be full to a three-dimensional region of size $x_{max} = 31$, $y_{max} = 8$, $z_{max} = 6$. The first iteration of the "Force to be Full" algorithm produced a distribution of resources with essentially no spatial pattern (Fig. 4b), but by the time the algorithm is finished, for only a moderate number of iterations ($n_{max} = 20$), the fit between the empirical structure function and that corresponding to the third-order Markov process is excellent (Fig. 4c). Since the "Force to be Full" algorithm generates the actual distribution of resources, we can consider that as well or, more informatively, cross sections of that distribution (Fig. 4d).

DISCUSSION

We believe that the structure function, the conditional probability of finding resource at a point as a function of the distance from a given point, is an intuitive (because of its focus on local conditions), usable and interpretable means to describe spatial pattern. Structure functions can be easily estimated in the field and provide sufficient information to generate patterns in multiple dimensions with the Force to be Full algorithm. Although we have focussed on structure functions corresponding to presence or absence of resource, more complicated cases work in the same fashion. For example, human krill fishers differentiate between "green" krill (those that have recently fed and for which there is little market value) and "pink" krill (those that have not recently fed and that have high market value). In that case, we need to introduce three structure functions: one conditioned on absence of krill at a point, one conditioned on the presence of green krill, and one conditioned on the presence of pink krill. Similarly, we can superimpose upon the presence or absence of resource a distribution of sizes. For example, we might use the structure function to describe whether clusters of fruit are present or not in a cell and then a different probability distribution for the number of fruit in the cluster.

We believe that the structure function is the appropriate tool to describe spatially distributed resources encountered by foragers. A fly minimizing distance travelled while ovipositing on rose hips that are clustered within bushes must decide how far to travel from points with and without fruit. Knowledge of the structure function and the costs of travel are sufficient information to compute the fitness of differing behaviors in this case (Mangel 1994). In dynamic situations, when fruits or resources are depleted by one or more foragers or are regenerated, the structure function can be used

both to describe the spatial consequences of the dynamics and to compute changing optimal behavioral responses. Such precise behavioral predictions can be tested either with direct observation of individuals or by careful measurement of the state of resources after foraging (Levin et al. 1977). For example, the pattern of parasitized and unparasitized fruits carries a record of the foraging behavior of tephritid fruit flies (Mangel 1994). Similarly, the nectar level in flowers, or even the pollination status of seeds, carry a record of the spatial foraging behavior of pollinators.

Our approach has antecedents in the literature of pattern recognition (Duda and Hart 1973), simulated annealing (Aarts and Korst 1989), Gibbs sampling (Ripley 1988:96), and neural computation (Amit 1989). Such methods can also be used to recognize and generate clustered spatial patterns by means of constructing an "energy" function related to the current spatial configuration and attempting to minimize that energy function. Usually some kind of Gaussian assumption is made, whereas none is required with our algorithm. In addition, in principle the method of the Force to be Full changes the entire pattern on each sweep through the algorithm, rather than changing the pattern incrementally.

There are several technical issues not addressed by the "Force to be Full" algorithm. There is no guarantee that the algorithm will approach an appropriate pattern. In the worst case, there might be no pattern with a given structure function. Although we avoided this situation by using structure functions created from a biological process, such infeasible functions do exist. For example, in one dimension, if the probability that a full point has a full neighbor is large [$p(1 \mid 1)$ is near 1], then the probability that a point two steps away from a full point is full cannot be too small since it must hold that $p(2 \mid 1) \geq p(1 \mid 1)^2 + [1 - p(1 \mid 1)]^2$. The complete conditions for consistency are given in Appendix 3. Even if patterns matching a given structure function do exist, we have been unable to demonstrate mathematically that the algorithm will find them. A demonstration of the convergence properties of the algorithm would be very instructive. Because we focus on a discrete description of space, the computational requirements (both in memory and time) depend more upon the size of cell that one considers than anything else. If the cells are very small, then it will be easy to exhaust the memory of a desktop microcomputer. However, the cell size should not be considered as fully arbitrary: it is determined in large part by the relevant spatial and temporal characteristics of the organism being studied.

Another related technical issue concerns the translation of distributions of nearest neighbor distances into structure functions. We showed how to do this in one dimension, but know of no technique to do so in multiple dimensions, even with additional assumptions more restrictive than that made in one dimension. If this translation proves impossible, the distribution of nearest neighbor distances probably cannot be used as anything more than a statistic.

Finally, consider the issue of when two structure functions differ significantly. Although this question can be phrased as one involving statistical considerations, it is our opinion that the answer is most relevant in the biological context. In this view, two habitats with potentially different structure functions are different if they lead to substantially different fitness consequences for the organism foraging in the two habitats, taking into account the relevant levels of behavioral plasticity. Mangel (1994) illustrates how the structure function can be connected to fitness.

ACKNOWLEDGMENTS

The work of F. R. Adler was supported by the Center for Population Biology at the University of California Davis. The work of M. Mangel was partially supported by NSF Grants OCE 90-16895 and BSR 91-17603. For comments on the manuscript, we thank Don Ludwig, Bill Morris, and an anonymous referee. F. R. Adler especially thanks Beate Nuernberger for stimulating him to think about the construction of clustered spatial patterns.

LITERATURE CITED

Aarts, E., and J. Korst. 1989. Simulated annealing and boltzman machines. John Wiley & Sons, New York, New York, USA.

Amit, D. J. 1989. Modeling brain function. Cambridge University Press, Cambridge, England.

Bell, W. J. 1991. Searching behaviour. Chapman and Hall, New York, New York, USA.

Butterworth, D. S., D. L. Borchers, and D. G. M. Miller. 1991. Some comments on the procedure for testing estimators of krill abundance which utilize survey data. Pages 191–204 *in* Selected Scientific Papers of the Scientific Committee for the Conservation of Antarctic Marine Living Resources. Commission for the Conservation of Antarctic Marine Living Resources, Hobart, Tasmania, Australia.

Casas, J. 1990. Multidimensional host distribution and nonrandom parasitism: a case study and a stochastic model. Ecology **71**:1893–1903.

Cressie, N. 1991. Statistics for spatial data. John Wiley & Sons, New York, New York, USA.

Duda, R. O., and P. E. Hart. 1973. Pattern classification and scene analysis. John Wiley & Sons, New York, New York, USA.

Gillespie, J. H. 1991. The causes of molecular evolution. Oxford University Press, Oxford, England.

Glick, N. 1978. Breaking records and breaking boards. American Mathematical Monthly, January 1978:1–26.

Karlin, S., and H. M. Taylor. 1975. A first course in stochastic processes. Second edition. Academic Press, New York, New York, USA.

Kauffman, S., and S. Levin. 1987. Towards a general theory of adaptive walks on rugged landscapes. Journal of Theoretical Biology **128**:11–45.

Levin, S. A., J. E. Levin, and R. T. Paine. 1977. Snowy owl predation on short-eared owls. Condor **79**:395.

Levin, S. A., and L. A. Segel. 1985. Pattern generation in space and aspect. SIAM Review **27**:45–67.

Mackas, D. L., K. L. Denman, and M. R. Abbott. 1985. Plankton patchiness: biology in the physical vernacular. Bulletin of Marine Science **37**:652–674.

Mangel, M. 1994. Spatial patterning in resource exploitation

and conservation. Proceedings of the Royal Society of London B **343**:93–98.

Miller, D. G. M., and I. Hampton. 1989. Krill aggregation characteristics: spatial distribution patterns from hydroacoustic observations. Polar Biology **10**:125–134.

Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. 1986. Numerical recipes. Cambridge University Press, Cambridge, England.

Ripley, B. D. 1988. Statistical inference for spatial processes. Cambridge University Press, Cambridge, England.

APPENDIX 1

First-order Markov Processes and Nearest Neighbor Distance

We here show that the structure function for the first-order Markov process can be recovered from the nearest neighbor distances of that process and Eq. 11.

The nearest neighbor distances for the first-order Markov process are

$$g(1) = p_1$$

$$g(d) = (1 - p_1)(1 - p_0)^{d-2}p_0 \qquad \text{for } d \geq 2.$$

Therefore, for $d \geq 3$, $g(d) = (1 - p_0)g(d - 1)$.

For convenience, we denote $p(r|1)$ by p_r. Then, we have

$$\begin{aligned} p_r &= \sum_{d=1}^{r} g(d)p_{r-d} \\ &= g(1)p_{r-1} + g(2)p_{r-2} + \sum_{d=3}^{r} g(d)p_{r-d} \\ &= g(1)p_{r-1} + g(2)p_{r-2} + \sum_{d=3}^{r} (1 - p_0)g(d - 1)p_{r-d} \\ &= g(1)p_{r-1} + g(2)p_{r-2} + \sum_{x=2}^{r-1} (1 - p_0)g(x)p_{r-x-1} \\ &= g(1)p_{r-1} + g(2)p_{r-2} + (1 - p_0)[p_{r-1} - g(1)p_{r-2}] \\ &= (1 - p_0 + p_1)p_{r-1} + (p_0 - p_1)p_{r-2} \\ &= p_{r-1} + (p_1 - p_0)(p_{r-1} - p_{r-2}). \end{aligned}$$

Therefore, $(p_r - p_{r-1}) = (p_1 - p_0)(p_{r-1} - p_{r-2})$, which, along with the initial condition $p_0 = 1$, matches the solution (Eq. 9) of Eq. 8.

APPENDIX 2

Second- and Third-Order Markov Processes and Their Structure Functions

Given the probabilities,

$$p_{ij} = \text{Prob}\{Z(r) = 1 \,|\, Z(r - 1) = i,\ Z(r - 2) = j\},$$

defining a second-order Markov process as in Eq. 20, we can generate a spatial distribution of resources according to the following algorithm.

Algorithm: Constructing a Second-Order Markov Resource Distribution

1) Set $Z(0) = 1$ and then draw a random number u uniformly distributed between 0 and 1. Then if $u < p_{00}/(p_{00} + p_{01})$, we set $Z(1) = 0$ and if $u \geq p_{00}/(p_{00} + p_{01})$, we set $Z(1) = 1$. Set $r_c = 1$.

2) Set $p_{\text{test}} = p_{Z(rc),\ Z(rc\ -\ 1)}$ and draw a random number u uniformly distributed between 0 and 1. Then if $u < p_{\text{test}}$, set $Z(r_c + 1) = 1$; otherwise set $Z(r_c + 1) = 0$.

3) Replace r_c by $r_c + 1$ and if the new value is less than $r_{\max}$, return to step 2.

Once we have generated the pattern $\{Z(r)\}$ in this manner, we can use Algorithm 1 to construct the structure function. A slight modification of this algorithm allows the construction of a pattern from a third-order Markov process.

APPENDIX 3

Consistency Conditions on Structure Functions

The following condition is modified very slightly from Karlin and Taylor (1975:504). Suppose that $\{X_n;\ n = 0, \pm 1, \pm 2, \ldots\}$ is a stochastic process on the integers with mean 0 and variance 1. The covariance function can be defined as

$$R(d) = E[X_n X_{n+d}].$$

A function R is the covariance function of a covariance stationary process if and only if it is positive semidefinite, that is, for all $k \geq 1$ and all real numbers $\alpha_1, \ldots, \alpha_k$

$$\sum_{i=1}^{k} \sum_{j=1}^{k} \alpha_i \alpha_j R(i - j) \geq 0.$$

To convert the Z_n to have mean 0 and variance 1, set

$$X_n = \frac{Z_n - p_a}{\sqrt{p_a(1 - p_a)}}.$$

Then

$$\begin{aligned} R(d) &= E[(Z_n - p_a)(Z_{n+d} - p_a)]/[p_a(1 - p_a)] \\ &= (E[Z_n Z_{n+d}] - p_a^2)/[p_a(1 - p_a)] \\ &= [p_a p(d|1) - p_a^2]/[p_a(1 - p_a)] \\ &= [p(d|1) - p_a]/(1 - p_a). \end{aligned}$$

The condition that $R(d)$ be positive semidefinite is then

$$\sum_{i=1}^{k} \sum_{j=1}^{k} \alpha_i \alpha_j [p(i - j|1) - p_a] \geq 0$$

or

$$\sum_{i=1}^{k} \sum_{j=1}^{k} \alpha_i \alpha_j p(i - j|1) \geq p_a \left(\sum_{i=1}^{k} \alpha_i \right)^2.$$

This is the required consistency condition.

APPENDIX 4

Timing of the "Force to be Full" Algorithm

The "Force to be Full" (FTBF) algorithm requires that for each cell containing resources, we cycle over all cells. Thus, if C is the number of cells, when C is large, the time to complete the algorithm will grow as C^2. To illustrate this effect, we ran the algorithm on a Macintosh Quadra 800 with the structure function shown in Fig. 4, cycling over four values of β, and for 20 iterations, for C ranging between 50 and 600 cells. We found that the time $T(C)$, measured in minutes, to complete the iterations fit the relationship ($r^2 = 1.00$):

$$T(C) = 7.1 + 0.0323C + 0.000542C^2. \quad \text{(A1)}$$

In addition to cell size, one must consider the rate of improvement of the algorithm. That is, we specify the number of iterates n_{max} to be used but each iteration does not lead to an improvement in the fit of the constructed structure function and the given structure function. Two analogies are helpful. First, each time a constructed pattern has a structure function that fits the given structure function better, we have "broken the previous record." The mathematics of record breaking is filled with nonintuitive results (Glick 1978). Alternatively, we can conceive of each new pattern representing a "mutation" that completely replaces the previous "best" pattern if the new structure function is in better agreement with the specified structure function than the current "best" structure function. Such mutational systems have the property that initial improvements occur rapidly but then subsequent improvements require increasing amounts of time (Kauffman and Levin 1987, Gillespie 1991). To illustrate this idea, we ran the FTBF algorithm with 100 cells, the structure function used in Fig. 4, and measured the number of iterations (a random variable) required to achieve 11 improvements. We then fit the data to find the iteration $I(k)$ on which the k^{th} improvement occurs is given by ($r^2 = 0.941$)

$$I(k) = 1.071 \times 10^{0.3256k}. \quad \text{(A.2)}$$

This result shows how the number of iterations needed to achieve a specified number of improvements grows exponentially.